Measurement of egress and ingress delays of PTP clocks

Christian Riesch, Cristian Marinescu, Manfred Rudigier
OMICRON electronics GmbH
Oberes Ried 1, 6833 Klaus, Austria
[christian.riesch|cristian.marinescu|manfred.rudigier]@omicron.at

***Abstract*—Distributed systems evolved to state-of-the-art solutions in many areas, like test, measurement and automation control systems. The protocol defined by the IEEE 1588 standard was designed to achieve clock synchronization among the different components of a distributed system using an unreliable communication network. Since any asymmetry between master and slave impacts the synchronization accuracy, PTP software has to compensate such asymmetries as much as possible. Our paper studies the effects of the hardware propagation delays on the synchronization. We present a setup that allows the direct measurement of these time delays and show results for a PTP clock that is generating the timestamps in the PHY and for a second one that is creating them on a MAC level. We discuss the characteristics of the time delays, and demonstrate the feasibility of the setup by using the measured compensation values in a synchronization experiment.**

***Index Terms*—IEEE 1588, hardware propagation delay, PTP, synchronization**

I. Introduction

While many test and measurement, real-time simulation or automation control systems become more and more complex, distributed systems evolved to be the state-of-the-art solution in these areas. Processing the distributed data and events requires their merging in a timely order, so that monitoring, analysis and measurement functions can be applied and according actions can be taken. This leads us to the necessity to have the same time base available throughout the whole system, meaning that all clocks of all devices must have approximately the same time [1]. Unfortunately, the characteristics of digital oscillators present a high dependency on factors like temperature, humidity and pressure [2]. Even though the clocks could be initialized in sync, because of the above mentioned hardware limitations, it becomes necessary to continuously adjust the clocks to keep them synchronous. This process is called synchronization [3].

In order to achieve the required synchronization, the devices forming the distributed system must either access timing signals from a common time source, or they have to synchronize their internal clocks in order to share a common time base [1]. The first method was extensively used in the past, as long as the devices were close together. The second method becomes necessary whenever the system is widely distributed and the costs of distributing the time signal exceed a certain limit.

The Precision Time Protocol (PTP) provided by the IEEE 1588 standard [4] was designed to fulfill synchronization among distributed devices, connected through a non-deterministic, multicast capable network. By achieving clock accuracies in the sub-microsecond (and, under special circumstances, even sub-nanosecond) range, IEEE 1588 enabled a wide range of applications in a distributed environment. One key requirement needed to reach such accuracies is the capability of the hardware to offer precise time-stamping support for the PTP packets. Nevertheless, many factors, such as oscillator stability, clock frequency, the propagation delay through the physical layer (PHY), or any other jitter source, have a negative impact on the synchronization accuracy. Under these circumstances, the goal to reach better accuracy explains the struggle for each and every improvement, even in terms of a couple of nanoseconds.

There are a couple of different time-stamping points possible in a PTP device (see Figure 1) [5], [6]. The timestamps can be generated in hardware (in the PHY or between PHY and Media Access Control (MAC)) or in software (inside the network driver or in the PTP software). To obtain a high accuracy of the timestamps with almost no jitter at all, it is necessary to generate the timestamps as close as possible to the communication wire, so we consider that the best approach is to have them generated in the PHY chip. The PTP stack will require an interface to collect the timestamps generated for incoming and outgoing PTP frames [5].

Our paper concentrates on the delays between the time-stamping point and the reference plane and how to reliably measure them. The paper is structured as follows: section II presents the motivation why knowing the egress and ingress delays through the hardware are important from a PTP synchronization perspective, elaborating on the background of

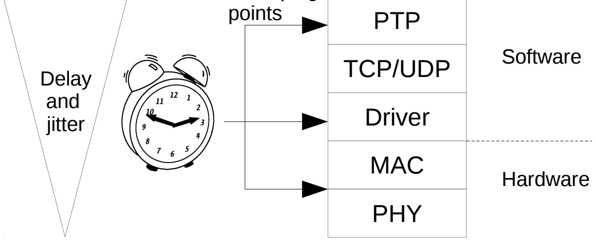


Fig. 1: Time-stamping points in a PTP node

the problem and on previous work. Section III presents a measurement setup, while section IV details the obtained results. Afterwards, the paper presents our conclusions.

II. Motivation, Background and Previous Work

For each sent and received event message, a time-stamp is generated in hardware. According to [4], clause 7.3.4.2, *the time-stamp shall be the time at which the event message time-stamp point passes the reference plane marking the boundary between the PTP node and the network.* Since timestamps are actually generated at a point removed from the reference plane, they are affected by ingress and egress latencies. But why are the egress and ingress delays important? Wouldn't it be possible to just neglect the couple of dozens of nanoseconds delay between the time-stamping point and the real frame start on the wire?

It is a well-known fact that PTP assumes symmetrical wire delays (the delay from master to slave is assumed to be equal to the delay from the slave to the master). Any delay difference in this direction is causing small errors that lead to worse synchronization results. In general, the active components on the path between master and slave, like switches, routers, but also the PHYs of the master and the slave, are responsible for such asymmetries. For direct connections, the major part of the asymmetry is caused by the two PHYs, and their egress and ingress delays will be probably different, since there is a good chance that, coming from different vendors, they are completely different (Figure 2). IEEE 1588-2008 already provides support for asymmetry compensation, systematic errors can be compensated, if the asymmetry is constant and the jitter is small. But in order to do so, it is necessary to know the propagation delays, or to be able to measure them, since the protocol does not offer means to determine asymmetries.

Our first concern was to find out how big the introduced inaccuracies would be, if we would decide to neglect the hardware egress and ingress delays. In order to analyze these synchronization inaccuracies, we decided to make two measurements, one in which the delays were not compensated, and one in which they were compensated (with already known values). For these measurements, we took a simple measurement setup into consideration (Figure 3): an OMICRON Lab OTMC 100 was used as PTP master, directly connected over 100Base-TX to a P2020RDB from Freescale, that was used as a PTP slave. Both master and slave carried out a Pulse Per Second (PPS) output, both PPS signals being analyzed by a Tektronix DPO4054B oscilloscope.

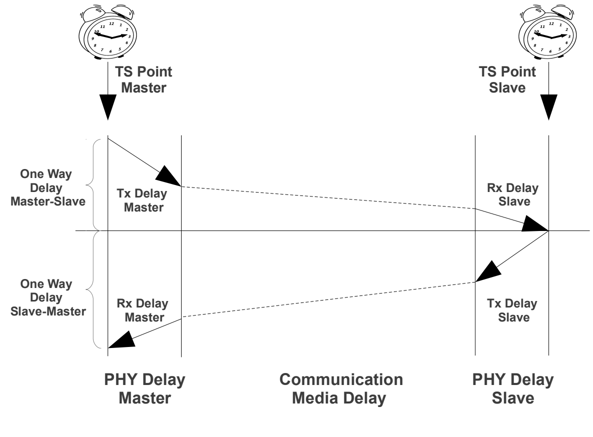


Fig. 2: Asymmetric propagation delay, partially caused by the hardware.

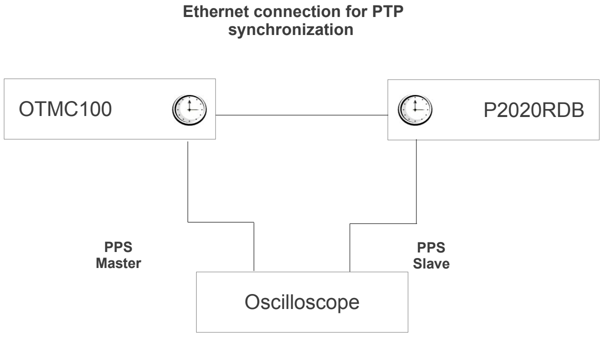


Fig. 3: Measurement setup for the analysis of the propagation delays.

The results of the experiment are shown in Figure 4. We noticed that for the uncompensated mode, even though directly connected, the slave had problems in synchronizing to the master with an accuracy better than +/-100 ns (which, in our experience should be the case for directly connected PTP nodes, when delays are compensated). We experienced a time difference between -876 ns and -333 ns between master and slave, with a mean value of -411 ns, and a median of -408 ns. We can conclude that no high precision synchronization is possible without taking the hardware delays and the introduced asymmetry into account.

One possible way to get the egress and ingress delays is to consult the technical data sheet provided by the hardware

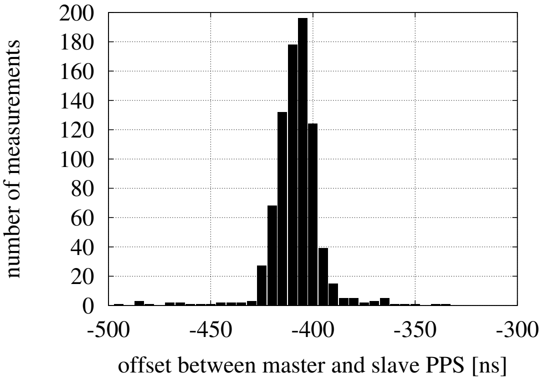


Fig. 4: Time delay between the PPS signals of the master and the slave clock. No compensation of the ingress and egress delays was done (mean value -411 ns, median -408 ns).

TABLE I: Adjustment values of the DP83640 PHYTER chip from National Semiconductor

Connection type	Direction	Adjustment values [ns]
100Base-TX	Transmit	0
	Receive	215
10Base-T	Transmit	95
	Receive	300

vendors. The delays can be found under different terms, depending on the vendor (e.g., adjustment value, latency): [7] presents adjustment values for the send path to account for *the delay from the time-stamp point to the wire* or for the receive path to compensate for *the delay from the wire to the time-stamp point*; [8] defines the latency of the PHY *from the point where the start of frame is detected at the SGMII interface (MAC) to the start of frame arriving at the outgoing pins (cable)*, the time-stamping point being the SGMII interface.

The data sheets that we have studied ([7], [8], and [9]) present different values for the propagation delays, for both directions ingress and egress. The values depend on the type of interface used (RGMII/SGMII/GMII), on the cable type connected (electrical or fiber), and also on the link speed (10/100/1000Mb). Some data sheets provide a minimum and a maximum value for the propagation delay, implying the existence of a jitter of the propagation delays. The PTP software that is expected to make the compensation needs to store these values and apply the appropriate average value, to correct each timestamp.

Table I presents typical values for the propagation delays of the DP83640 PHYTER from National Semiconductor, obtained from the data sheet [7]. Unfortunately, the vendor neither specifies how the values were obtained, nor under which circumstances. Table II shows some typical values for the VSC85xx and VSC86xx PHY families from Vitesse, obtained also from the data sheet [8]. The vendor specifies that the presented values were not measured, but simulated.

Since we wanted to verify the provided compensation values, we started to study possibilities how to measure the egress and ingress delays. Under these circumstances we looked for any other research in this area, and found but a few articles about this topic. [10] studies the propagation delay variation on 100Base-TX Ethernet PHY chips and presents a measurement setup for how to measure the delay. The problem with the presented approach was to find out where the Ethernet frame would start, so it was necessary to introduce some limitations to solve the problem. This is why, unfortunately, the method works only for 100Mb half-duplex connections with auto negotiation disabled. Further, [11] analyzes the transmit and receive delay of the PHYs, and finds out that they are vendor specific and are different by approximately a factor of 3. Since these delays highly depend on connection speed, cable type and interface, we argue that a method to reliably measure them, working under any circumstances, is needed.

TABLE II: Typical latencies for some Vitesse PHYs of the VSC85xx and VSC86xx families

Connection type	Direction	Min. latency [ns]	Max. latency [ns]
SGMII, 1000BaseT	Transmit	120	132
	Receive	260	292
RGMII, 100BaseT	Transmit	170	200
	Receive	320	380

III. Measurement setup

Figure 5 depicts a typical setup for an embedded system, comprising of a media access controller and the Ethernet PHY. In this setup, the timestamps t'_e and t'_i are taken by the PHY and are affected by ingress and egress latencies Δt_e and Δt_i. To compensate for these delays the corrections

$$\begin{aligned} t_\mathrm{e} &= t'_\mathrm{e} + \Delta t_\mathrm{e} \qquad \text{and} \\ t_\mathrm{i} &= t'_\mathrm{i} - \Delta t_\mathrm{i} \end{aligned} \tag{1}$$

must be applied [4].

The PHY shown in Figure 5 offers the possibility to time-stamp external events like a rising edge on a certain input pin of the device. The generation of such a time-stamp is affected by a time delay Δt_ev. Furthermore, the PHY can generate trigger signals like a rising edge at a programmed time, or a PPS signal. Naturally, there will be a delay between the time at which the internal clock/counter of the PHY exceeds the programmed value and the time at which the pulse actually egresses the output pin of the device. This delay is labeled Δt_tr in Figure 5.

To correctly apply the corrections of equation (1) (and similar corrections for external events and triggers), the time delays must be known. Our first attempt was to evaluate the propagation delays by using a simple loop-back cable. In fact, by sending a PTP packet and immediately receiving the same packet back over the loop-back cable, we received two timestamps, t'_e and t'_i. The difference between them is the sum of the egress delay, the ingress delay and the propagation through the loop-back cable. But since we used a short loop-back cable (with a propagation delay well under 1 ns), we could neglect the influence of the cable. Unfortunately, this simple method

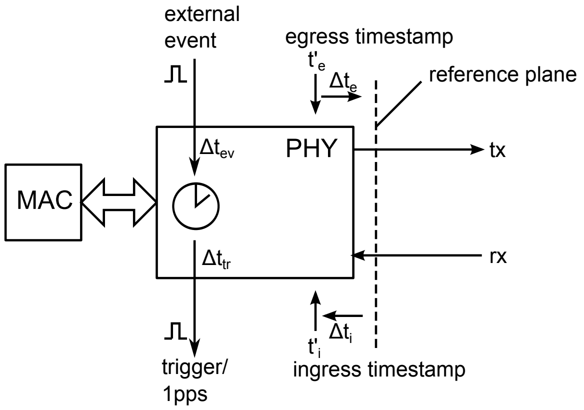


Fig. 5: Typical setup for an embedded system employing a PHY with PTP time-stamping capability. The figure shows the time delays that affect the generated timestamps.

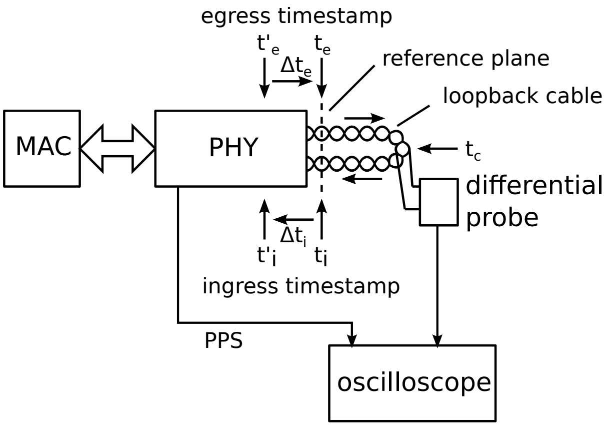


Fig. 6: Measurement setup

helped us to find out the sum $\Delta t_\mathrm{e} + \Delta t_\mathrm{i}$ of both delays, but since they are not equal (they differ approximately by a factor of 3), it was not possible to find out the corresponding values. Therefore, we developed a measurement setup that allows the direct measurement of both Δt_e and Δt_i.

Figure 6 shows the setup that was used to measure the PHY delays. For our experiments, we used a Tektronix DPO4054B oscilloscope that was equipped with the *DPO4ENET* option. This option enables the oscilloscope to trigger on events received on the Ethernet line (e.g., the start/end of an Ethernet frame). We connected a loop-back cable to the Ethernet port of the device under test (DUT), while the oscilloscope was connected to the data lines of this cable using a differential probe. In a first attempt, the scope was set to trigger on the start of frame delimiter of a 100BaseTX Ethernet frame. However, these settings did not work reliably, and therefore we used the end of the MAC addresses as trigger point. A correction was applied to the measurement results to comply with the timestamp point definition of clause 7.3.4.1 [4]. On the DUT we ran a test program that transmitted a PTP sync packet over the monitored Ethernet port. The time-stamp t'_e, that was generated by the PHY on egress, was printed to the console:

$$t'_\mathrm{e} = T'_\mathrm{e,sec} + T'_\mathrm{e,nsec} \cdot 10^{-9}, \tag{2}$$

where $T'_\mathrm{e,sec}$ is the seconds and $T'_\mathrm{e,nsec}$ the nanoseconds part of the time-stamp. Both $T'_\mathrm{e,sec}$ and $T'_\mathrm{e,nsec}$ have positive integer values. The same sync packet was received back over the loop-back cable by the test program and the ingress timestamp t'_i was also printed to the console.

For the measurement of the delays Δt_e and Δt_i we had to determine the egress and ingress timestamps at the reference plane, t_e and t_i (see Figures 5 and 6). Since the used loop-back cable was very short, we neglected the delay (well below 1 ns) introduced by the cable. We also defined the loop-back cable to be the reference plane and labeled the time at which the oscilloscope detected the packet on the cable t_c:

$$t_\mathrm{c} = t_\mathrm{e} = t_\mathrm{i}. \tag{3}$$

To align the oscilloscope measurements with the clock inside the PHY, a PPS signal was generated by the PHY and was connected to the second channel of the scope. A positive edge on the PPS signal marks the beginning of a full second (Figure 7). The delay between the positive edge of the PPS signal and the timestamp point of the sync packet is measured using the scope's waveform search functions. In our measurements, the oscilloscope was configured to acquire 10^6 measurements at 1.25 Gigasamples per second. Thus, the time delay can be determined with a resolution of 800 ps. The time interval covered by the oscilloscope measurement is $800\,\mathrm{ps} \cdot 10^6 = 800\,\mu\mathrm{s}$. Consequently, the test program must ensure that the packet is sent within a time frame of 800 µs around the PPS pulse by polling the PTP time and sending the sync packet in the appropriate time window. Otherwise, the PPS pulse would lie outside the measurement range of the oscilloscope.

The time-stamp t_c is obtained from the delay measurement of the oscilloscope (Δt_scope). Since the PPS pulse is generated at the begin of a full second and the data packet was timestamped at $t'_\mathrm{e} = T'_\mathrm{e,sec} + T'_\mathrm{e,nsec} \cdot 10^{-9}$, that is a point in time that is within 800 µs around the PPS pulse, we conclude that the absolute time at which the PPS pulse was generated is

$$T_\mathrm{c,sec} = \begin{cases} T'_\mathrm{e,sec} & \text{for } T'_\mathrm{e,nsec} < 500\,000\,000 \\ T'_\mathrm{e,sec} + 1 & \text{for } T'_\mathrm{e,nsec} \geq 500\,000\,000 \end{cases}. \tag{4}$$

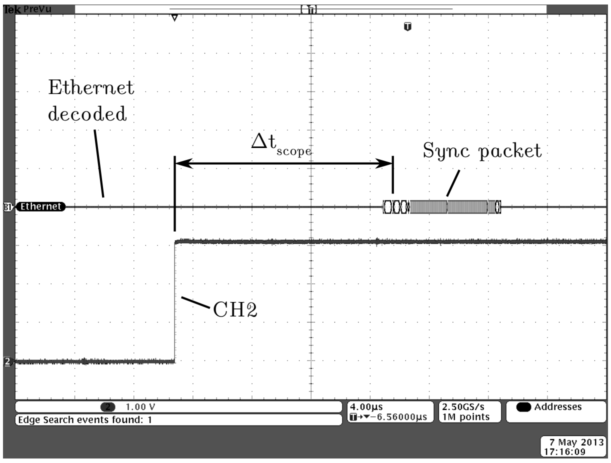
Adding the delay measurement result of the oscilloscope,

$$T_\mathrm{c,nsec} = \Delta t_\mathrm{scope} \cdot 10^9, \tag{5}$$

yields the time-stamp of the data packet on the cable,

$$t_\mathrm{c} = T_\mathrm{c,sec} + T_\mathrm{c,nsec} \cdot 10^{-9}. \tag{6}$$

For a correct measurement of the delay between the PPS pulse and the data packet, the delays caused by the oscilloscope probes and cables must be matched. Furthermore, the delay Δt_tr (see Figure 5) affects the result. However, we



2 1.00 V
Edge Search events found: 1
4.00µs
T→▼−6.56000µs
2.50GS/s
1M points
● Addresses
7 May 2013
17:16:09

Fig. 7: Triggering on a sync packet with the oscilloscope. Channel 2 of the scope is connected to the PPS signal generated by the DUT. The positive edge marks the begin of a full second. The oscilloscope's waveform search function is used to measure Δt_scope.

TABLE III: Measurement results for the egress and ingress delays for 100Base-TX Ethernet.

DUT	Samples	Direction		min. [ns]	max. [ns]	mean [ns]	median [ns]
DP83640 PHY	473	egress	Δt_e	-19.8	-16.6	-18.6	-18.8
		ingress	Δt_i	232.6	235.8	234.6	234.8
P2020 MAC	340	egress	Δt_e	-215.6	-169.0	-191.5	-191.5
		ingress	Δt_i	874.4	965.2	882.8	881.4

expected this delay to be very small and therefore defined $\Delta t_\mathrm{tr} \overset{!}{=} 0$. Setting Δt_tr to zero will cause a small time offset of the internal PTP clock of the DUT to the reference time. This offset is acceptable, since the timestamps and trigger pulses generated by the device will remain correct (the correction values determined in the measurement will also correct this clock offset). The offset will only become effective if the current time of the clock is read from software, but in this case, software latencies, much larger than the introduced error, will be present [12]. Furthermore, the time bases of the DUTs and the oscilloscope must be syntonized to a frequency reference. However, due to the precise internal time bases of the devices and the short measurement duration (below 800 µs) the resulting error was estimated to be in the sub-nanoseconds range. Therefore, no syntonization was done in the measurements presented below.

Both oscilloscope and DUT were controlled from a computer over Ethernet and RS232, respectively. The computer first arms the oscilloscope's trigger and then commands the DUT to send a PTP sync message. It then reads the timestamps t'_e and t'_i and the time delay Δt_tr from the devices. Equations (4) and (5) yields the timestamp of the data packet on the cable. From the timestamps, the egress and ingress delays can be calculated as follows:

$$\begin{aligned} \Delta t_\mathrm{e} &= t_\mathrm{c} - t'_\mathrm{e}, \\ \Delta t_\mathrm{i} &= t'_\mathrm{i} - t_\mathrm{c}. \end{aligned} \tag{7}$$

IV. Measurements

The measurements of Δt_i and Δt_e were carried out with two different DUTs. The first device is a custom board based on the National Semiconductor PHY DP83640 with time-stamping capabilities in the PHY and a Texas Instruments AM1808 system-on-chip. The second device is a Freescale P2020RDB board, on which the PTP timestamps and the PPS signal (see Figure 6) are generated by the MAC. Both devices are running Linux, the test program that sends and receives the data packets uses the PTP hardware clock infrastructure for the Linux kernel [13]. Therefore, the code runs on both platforms without any modifications. The loop-back cable/differential probe was connected to the Ethernet port of the DUT. On the devices, all programs that communicate over the network were terminated. Thus, we ensured that only PTP event messages from the test program were transmitted over the monitored network port.

The results of the experiments are listed in Table III. For the device with the DP83640 PHY we note that the results

$$\Delta t_\mathrm{e} + \Delta t_\mathrm{i} = -18.6\,\mathrm{ns} + 234.6\,\mathrm{ns} = 216\,\mathrm{ns} \tag{8}$$

are in agreement with the values from the data sheet (Table I). However, the egress delay Δt_e is negative. This means that the sync packets appears on the wire 18.6 ns before the time-stamp is taken. The reasons for this may be either hidden in the time-stamping logic of the device, or it may be caused by defining the PPS signal output delay Δt_tr to be zero. Since the time-stamping of the data packets is done in the PHY, very close to the physical media, the time delays of the DP83640 are very stable: the jitter introduced to the timestamps is very low (see the minimum and maximum values of the measurements in Table III).

The measurement results for the P2020RDB show a different characteristic. The total delay over the loop-back cable is much higher, since the time-stamping is done in the MAC. Hence, the propagation delay of the data packets through the entire PHY contributes to Δt_e and Δt_i. Furthermore, the delays are subject to a large jitter which deteriorates the time-stamping performance of the setup (see Figure 8).

Very similar to the DP83640 measurement, the egress delay

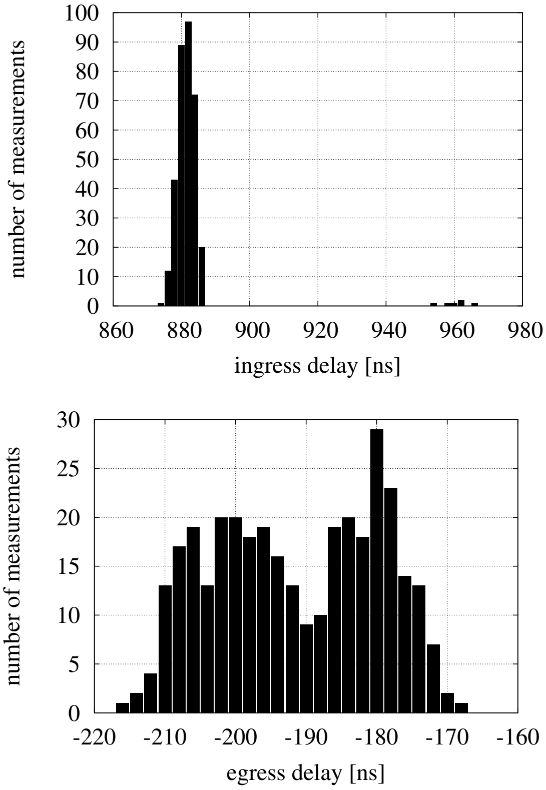


Fig. 8: Ingress delays Δt_i and egress delays Δt_e measured with the P2020RDB.

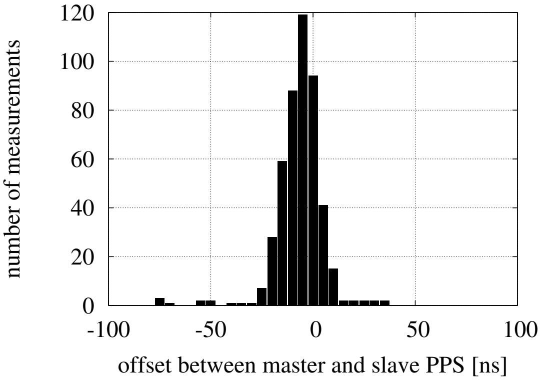


Fig. 9: Synchronization accuracy of the two devices under test with the ingress and egress delay compensation values applied.

Δt_e is negative, but the absolute value is much higher than for the PHY. It is hard to believe that a delay of this magnitude is caused by the internal logic of the device. We can only assume that some (over-)compensation of the PHY delays is done by the device before the timestamps are made available to the software.

To validate the egress and ingress delays obtained from the measurements, the experiment for the analysis of the hardware propagation delays (Figure 3) was repeated. However, this time the PTP stacks running on both devices were set up to perform the delay compensation given in equation (1). The ingress and egress delay correction values were 235 ns and -19 ns for the DP83640-based device, and 883 ns and -192 ns for the P2020RDB, respectively. The synchronization results (delays between the PPS pulses of both devices) are shown in Figure 9. The mean value of the synchronization error was reduced to -6.6 ns (median -5.8 ns). This remaining clock deviation is in the range of the time-stamping resolution of the DUTs (8 ns for the DP83640 and 5 ns for the P2020), which may cause a systematic error in the measurement of the egress and ingress delays. The results show a jitter characteristic, caused by the PHY delays of the P2020 device. Using suitable filtering, the synchronization performance could be further improved.

V. Conclusions

Time synchronization is one of the key requirements for a distributed system. For a precise distribution of time using the PTP protocol, each device must compensate the asymmetry it introduces to the network path. Therefore, the ingress and egress delays of PTP clocks have to be well known.

An automated measurement setup using an oscilloscope with Ethernet decoding features can precisely decode and time-stamp a PTP data packet on the network. Together with the data obtained from the DUT, the time delays of interest can be obtained. Applying the results as correction values allows a precise time synchronization of the devices. The feasibility of the measurement setup and the improved synchronization performance have been demonstrated successfully.

The measurement highly depends on the capabilities of the oscilloscope and its used option. Since the DPO4ENET option only supports 10Base-T and 100Base-TX Ethernet, our measurement setup can be used only for 10Base-T and 100Base-TX full-duplex connections. We had no possibility to verify our concept for 1Gb connections. Since the measurement range of the oscilloscope was 800 µs in our experiments, the PTP packet had to be sent within the time window around the edge of the PPS signal. This requirement was fulfilled by running a customized test software on the DUT. Therefore, the measurement setup cannot be used for verifying off-the-shelf devices that do not offer the possibility to run custom software. However, by increasing the number of data points that are acquired by the oscilloscope (max. $20 \cdot 10^6$), reducing the scope's sample rate (at the cost of lower time-stamping resolution), and increasing the sync rate of the DUT, we believe that a modified setup could be used for such a verification purpose.

Acknowledgements

The authors would like to thank their colleagues Thomas Ganahl, Michael Jochum, and Mark Marshall for their help and support during the work on this paper.

References

[1] A. Tan. IEEE 1588 and the need for Ethernet clock synchronisation. National Semiconductor Corporation. [Online]. Available: http://www.iebmedia.com/index.php?id=5543&parentid=63&themeid=255&hft=40&showdetail=true&bb=1&PHPSESSID=lj4k6tqnshleppn1ffi2762ar2

[2] D. W. Allan *et al.*, "Precision oscillators: Dependence of frequency on temperature, humidity and pressure," in *Proceedings of the 1992 IEEE Frequency Control Symposium*, 1992, report of Working Group 3 of the IEEE SCC27 Committee.

[3] (2008) Introduction to distributed clock synchronization and the IEEE 1588 precision time protocol. [Online]. Available: http://zone.ni.com/devzone/cda/tut/p/id/2822

[4] *Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*, IEEE Std. 1588-2008, 2008.

[5] H. Weibel and D. Bechaz, "IEEE 1588 implementation and performance of time stamping techniques," in *2004 Conference on IEEE 1588*, Sep.28 2004.

[6] J. Wu, "Implementation of IEEE 1588 precision time protocol on embedded processors," Analog Devices Inc., Tech. Rep. [Online]. Available: http://www.ieeeboston.org/publications/society_presentations/Implementation_of_IEEE_1588_Protocol_on_Embedded_Processors_Pub_Jiang_Wu.pdf

[7] "National Semiconductor Ethernet PHYTER software development guide," National Semiconductor Corporation, Tech. Rep., 2008. [Online]. Available: http://www.datasheetarchive.com/DP83640-datasheet.html

[8] "Gigabit Ethernet PHY device latency report," Vitesse, Tech. Rep., 2010.

[9] "AN-1507 DP83848 and DP83849 100Mb data latency," Texas Instruments, Tech. Rep., 2013. [Online]. Available: http://www.ti.com/lit/an/snla084b/snla084b.pdf

[10] J. Blattner and H. Weibel, "Study on propagation delay variation of 100Base-TX Ethernet PHY chips - supplement for conference contribution entitled *PHYs and Symmetrical Propagation Delay*," in *2004 Conference on IEEE 1588*, Sep.28 2004.

[11] T. Mueller, A. Ockert, and H. Weibel, "PHYs and symmetrical propagation delay," in *2004 Conference on IEEE 1588*, Sep.28 2004.

[12] R. Cochran, C. Marinescu, and C. Riesch, "Synchronizing the Linux system time to a PTP hardware clock," in *Precision Clock Synchronization for Measurement Control and Communication (ISPCS), 2011 International IEEE Symposium on*, Sep. 2011, pp. 87 – 92.

[13] R. Cochran and C. Marinescu, "Design and implementation of a PTP clock infrastructure for the Linux kernel," in *Precision Clock Synchronization for Measurement Control and Communication (ISPCS), 2010 International IEEE Symposium on*, Sep. 27–Oct. 1, 2010, pp. 116 –121.